ESAME DI FONDAMENTI DI INFORMATICA T-2 del 5/7/2022
Proff. E. Denti – R. Calegari – A. Molesini
Tempo a disposizione: 3h30

NOME PROGETTO ECLIPSE: CognomeNome-matricola (es. RossiMario-0000123456)
NOME CARTELLA PROGETTO: CognomeNome-matricola (es. RossiMario-0000123456)
NOME ZIP DA CONSEGNARE: CognomeNome-matricola.zip (es. RossiMario-0000123456.zip)
NOME JAR DA CONSEGNARE: CognomeNome-matricola.jar (es. RossiMario-0000123456.jar)

Si devono consegnare DUE FILE: *l'intero progetto Eclipse* e *il JAR eseguibile*

Si ricorda che compiti *non compilabili* o *palesemente lontani da 18/30* NON SARANNO CORRETTI e causeranno la verbalizzazione del giudizio "RESPINTO"

Si vuole sviluppare un'applicazione che consenta di monitorare la carriera universitaria di studenti, mostrando media pesata e crediti acquisiti, oltre a effettuare controlli generali sulla correttezza della carriera stessa.

DESCRIZIONE DEL DOMINIO DEL PROBLEMA

Un'***attività formativa*** rappresenta un insegnamento universitario caratterizzato da *numero di codice* (univoco), *denominazione* (che può contenere spazi) e *numero di crediti* (non necessariamente intero).

Quando, in una certa *data,* lo studente sostiene un ***esame***, all'attività formativa viene associato un ***voto,*** che può essere o un *numero fra 18 e 30* (quest'ultimo eventualmente con lode) o un giudizio di *idoneità*; in caso di esito negativo, viene attribuito uno dei giudizi *ritirato* o *respinto*. Ogni esame superato comporta l'acquisizione dei relativi *crediti*. È possibile ri-sostenere un esame solo se in precedenza l'esito è stato negativo, mentre non è consentito ripetere un esame già sostenuto con esito positivo.

Al termine della carriera, lo studente deve sostenere la ***prova finale***: essa può quindi essere sostenuta solo *in data successiva* ad ogni altro esame.

In qualunque momento la carriera dello studente è quindi caratterizzata da:
- **Lista degli esami sostenuti** (sia con esito positivo che con esito negativo)
- **Crediti acquisiti** (concorrono all'acquisizione dei crediti solo gli esami superati con *esito positivo*)
- **Media pesata** (concorrono all'acquisizione dei crediti solo gli esami con voto, superati con *esito positivo*)

La media pesata si calcola secondo la nota formula MP $= \frac{\sum v_i * p_i}{\sum p_i}$, essendo v_i i voti e p_i i pesi (crediti) degli esami.

Una serie di file di testo (i cui nomi non sono specificati né rilevanti) descrive possibili carriere di studenti**,** nel formato descritto più oltre.

TEMPO STIMATO PER SVOLGERE L'INTERO COMPITO: 2h15 – 3h

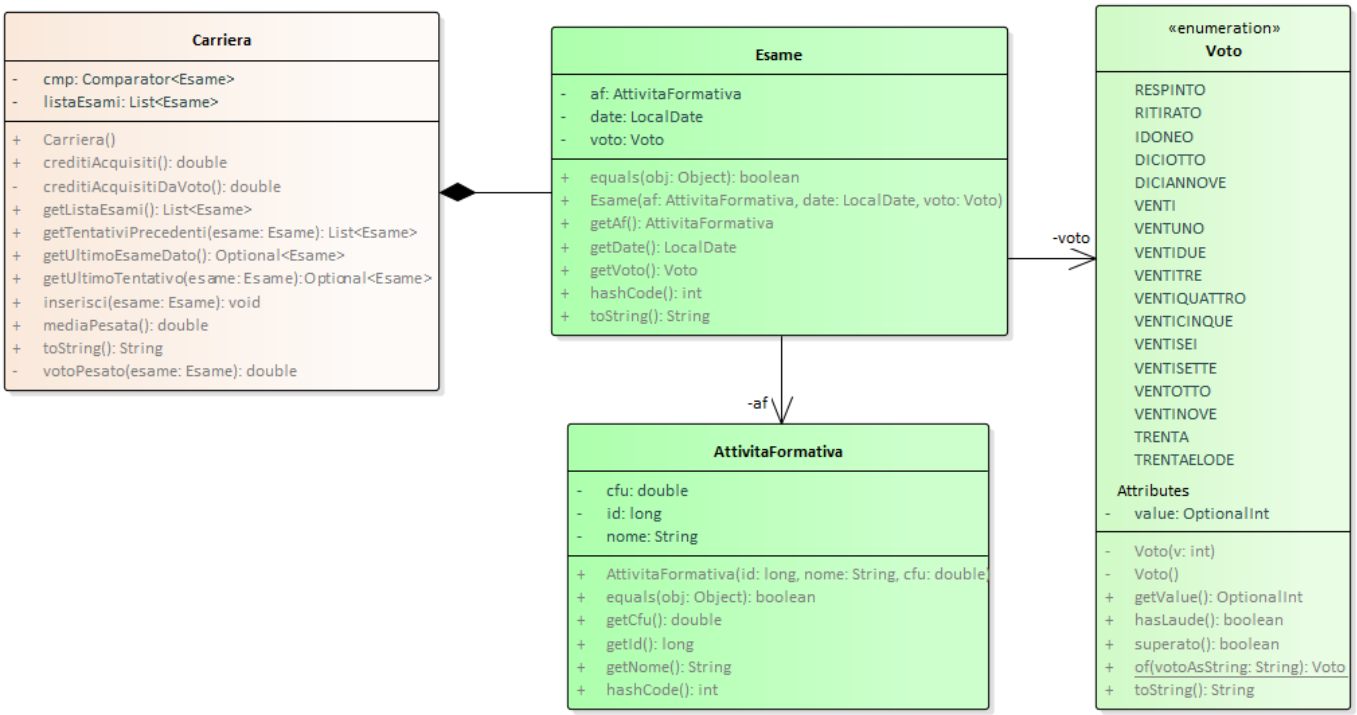
PARTE 1 – Modello dei dati: Punti 12 [TEMPO STIMATO: 55-70 minuti]
PARTE 2 – Persistenza: Punti 11 [TEMPO STIMATO: 50-70 minuti]
PARTE 3 – Grafica: Punti 7 [TEMPO STIMATO: 30-40 minuti]

JAVAFX – Parametri run configuration nei LAB

```
--module-path "C:\applicativi\moduli\javafx-sdk-17.0.2\lib"
--add-modules javafx.controls
```

Parte 1 – Modello dei dati (punti: 12)

Package: mediaesami.model **[TEMPO STIMATO: 55-70 minuti]**



SEMANTICA:

a) la classe ***AttivitaFormativa*** (fornito) descrive un'attività formativa come da dominio del problema

b) l'enumerativo ***Voto*** (fornito) rappresenta i possibili valori di voto o giudizio, come da dominio del problema; in particolare, sono definite sia costanti enumerative per ogni voto fra 18 e 30L, associate al corrispondente valore numerico (per 30L il valore è 30), nonché costanti enumerative per i tre giudizi respinto, ritirato e idoneo, non associate ad alcun valore numerico. Il metodo *getValue* estrae il valore intero associato, se esistente, mentre i due metodi *hasLaude* e *superato* sono veri rispettivamente nel solo caso del 30L, e per i voti positivi, ossia diversi da *ritirato* o *respinto*. Il factory method statico *of* restituisce la giusta istanza dell'enumerativo in base alla stringa ricevuta, che può essere o il valore numerico fra "18" e "30", oppure una delle stringhe "30L", "ID", "RT", "RE". Dualmente, un'apposita *toString* emette la stringa appropriata per ogni valore dell'enumerativo.

c) La classe ***Esame*** (fornito) rappresenta un esame, che associa a un'attività formativa un voto ottenuto in una ben precisa data. Il costruttore riceve tali dati, che sono poi recuperabili tramite gli appositi *accessor*; idonee *toString*, *equals*, *hashcode* completano l'implementazione.

d) la classe ***Carriera* (da implementare)** mantiene al proprio interno la lista degli esami sostenuti, ordinata per data crescente e in subordine per codice univoco dell'attività formativa. Il costruttore crea una carriera inizialmente vuota, che verrà poi riempita via via tramite il metodo *inserisci*. Più precisamente:

- il metodo *inserisci* inserisce un esame in carriera, a condizione che un esame per la stessa attività formativa non sia già stato sostenuto in precedenza con esito positivo (altrimenti, deve lanciare ***IllegalArgumentException*** con opportuno messaggio d'errore); deve inoltre, ovviamente, verificare che l'argomento ricevuto non sia nullo, lanciando in tal caso ***IllegalArgumentException*** con apposito messaggio d'errore. Nel solo caso in cui l'esame da inserire sia la prova finale (aspetto verificabile unicamente dalla descrizione, che in tal caso conterrà la dizione "PROVA FINALE"),

deve altresì *verificare che la data della stessa sia posteriore a quella di ogni altro esame in carriera* – anche in questo caso, lanciando ***IllegalArgumentException*** con adeguato messaggio

- il metodo *getListaEsami* restituisce semplicemente la lista ordinata degli esami attualmente presenti in carriera
- il metodo *getTentativiPrecedenti* restituisce la lista ordinata dei soli esami (con esito ovviamente negativo) sostenuti in precedenza per la stessa attività formativa dell'esame ricevuto come argomento (se non ce ne sono, restituisce una lista vuota)
- il metodo *getUltimoTentativo* restituisce, se esiste (per questo il tipo di ritorno è un ***Optional***), il più recente degli esami (con esito negativo) sostenuti in precedenza per la stessa attività formativa dell'esame ricevuto come argomento
- il metodo *creditiAcquisiti* restituisce la somma dei crediti acquisiti da esami superati con esito positivo (sia con voto numerico, sia con giudizio di idoneità)
- il metodo *creditiAcquisitiDaVoto* restituisce la somma dei crediti acquisiti da esami superati con esito positivo con solo voto numerico (utile per calcolare la media pesata)
- il metodo *mediaPesata* restituisce la media pesata dei voti (numerici) fin qui ottenuti
- un'apposita *toString* emette una stringa corrispondente alla lista di tutti gli esami (superati o meno), separati da "a capo", rispettando l'ordine di lista.

Parte 2 – Persistenza (punti: 11)

Package: mediaesami.persistence **[TEMPO STIMATO: 50-70 minuti]**

Sono forniti un certo numero di file di testo, tutti strutturati secondo il medesimo formato (possono contenere righe vuote). L'applicazione principale provvede da sé a recuperarne l'elenco dalla directory corrente e richiamarne ciclicamente la lettura: pertanto il candidato deve occuparsi unicamente di implementare il classico reader per un singolo file.

Ogni riga è organizzata in una serie di elementi separati da una o più tabulazioni. Tre di essi sono sempre presenti:

- codice identificativo dell'attività formativa (un intero long)
- denominazione dell'attività formativa (può contenere spazi e ogni altro carattere diverso da tabulazione)
- numero di crediti (un valore reale, scritto in formato italiano, con la virgola come separatore decimale)

Solo se l'esame è stato sostenuto, sono presenti due ulteriori elementi:

- data in cui l'esame è stato sostenuto (nel particolare formato GG/MM/AAAA)
  NB: tale formato non è fra quelli standard previsti dai formattatori, in quanto il formato SHORT prevede l'anno su due sole cifre, mentre i formati che prevedono l'anno su quattro cifre includono anche altri elementi, qui non presenti. Si suggerisce un attento studio della classe ***Esame*** per trarre ispirazione.. ☺
- voto (un numero fra 18 e 30, oppure una delle sigle 30L, ID, RT, RE rispettivamente per trenta e lode, idoneo, ritirato, respinto)

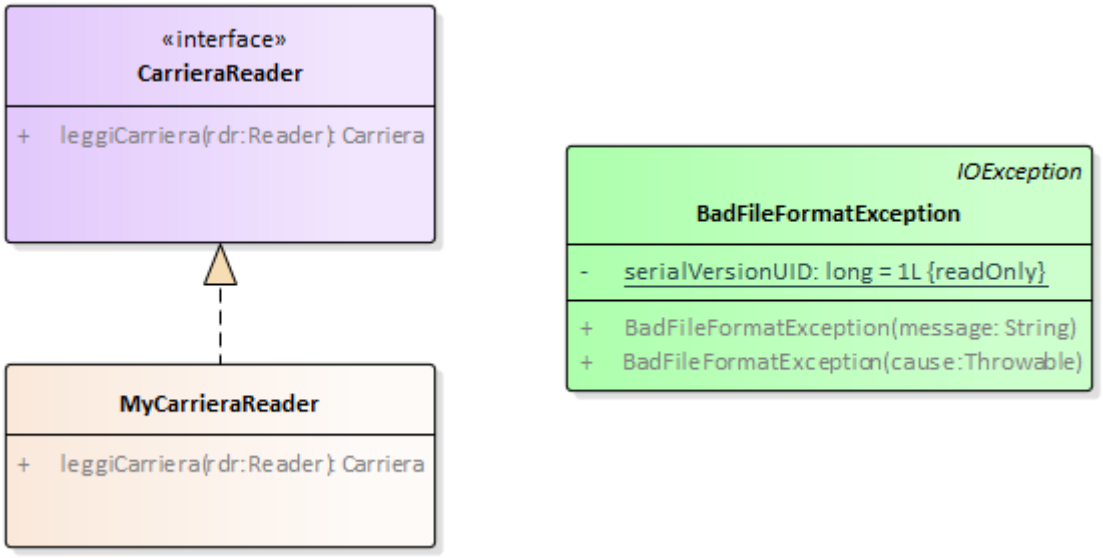
ESEMPIO:

```
27991    ANALISI MATEMATICA T-1                  9,0    12/1/2020    RT
27991    ANALISI MATEMATICA T-1                  9,0    10/2/2020    22
28004    FONDAMENTI DI INFORMATICA T-1           12,0   13/2/2020    24
29228    GEOMETRIA E ALGEBRA T                   6,0    18/1/2020    26
26337    LINGUA INGLESE B-2                      6,0    18/6/2020    ID
27993    ANALISI MATEMATICA T-2                  6,0    10/6/2020    RE
27993    ANALISI MATEMATICA T-2                  6,0    02/7/2020    RT
27993    ANALISI MATEMATICA T-2                  6,0    22/7/2020    23
28006    FONDAMENTI DI INFORMATICA T-2           12,0   21/7/2020    27
28011    RETI LOGICHE T                          6,0    22/2/2022    22

28012    CALCOLATORI ELETTRONICI T               6,0    22/1/2021    RT
28012    CALCOLATORI ELETTRONICI T               6,0    22/2/2021    20
```

```
30780    FISICA GENERALE T                          9,0    12/2/2021    25
28032    MATEMATICA APPLICATA T                     6,0    02/2/2021    24
28027    SISTEMI INFORMATIVI T                      9,0    03/6/2021    28
28030    ECONOMIA E ORGANIZZAZIONE AZIENDALE T      6,0    02/7/2021    RE
28030    ECONOMIA E ORGANIZZAZIONE AZIENDALE T      6,0    22/7/2021    24
28029    ELETTROTECNICA T                           6,0    02/9/2021    26
28014    FONDAMENTI DI TELECOMUNICAZIONI T          9,0    15/9/2021    30
28020    SISTEMI OPERATIVI T                        9,0    12/1/2022    24

28015    CONTROLLI AUTOMATICI T                     9,0    13/1/2021    25
28016    ELETTRONICA T                              6,0    10/2/2021    22
28024    RETI DI CALCOLATORI T                      9,0    05/2/2021    23
28659    TECNOLOGIE WEB T                           9,0    12/6/2021    25
28021    INGEGNERIA DEL SOFTWARE T                  9,0    24/6/2021    27
17268    PROVA FINALE                               3,0
28074    TIROCINIO T                                6,0    27/9/2021    ID
88324    AMMINISTRAZIONE DI SISTEMI T               6,0    13/7/2021    29
88325    LABORATORIO DI SICUREZZA INFORMATICA T     6,0
```



SEMANTICA:

a) L'eccezione ***BadFileFormatException*** (fornita) esprime l'idea di file formattato in modo scorretto

b) L'interfaccia ***CarrieraReader*** (fornita) dichiara il metodo *leggiCarriera,* che legge da un *Reader* (ricevuto come argomento) i dati di una carriera, configurando e restituendo l'opportuno oggetto ***Carriera.***
IMPORTANTE: poiché la ***Carriera*** è composta di ***Esami***, le righe contenenti la sola descrizione di attività formative, ossia quelle senza data e voto, devono essere comunque verificate a livello di formato ma ignorate per quanto riguarda l'inserimento in carriera, in quanto non descrivono un esame sostenuto.

c) La classe ***MyCarrieraReader*** (**da realizzare**) implementa ***CarrieraReader***: non prevede costruttori, si limita a implementare il metodo *leggiCarriera* come sopra specificato. In caso di problemi di I/O deve essere propagata l'opportuna ***IOException,*** mentre in caso di ***Reader*** nullo o altri problemi di formato dei file deve essere lanciata una opportuna ***BadFileFormatException***, il cui messaggio dettagli l'accaduto.
In particolare, il reader deve verificare, lanciando ***BadFileFormatException*** in caso contrario, che:

   i. la riga contenga tre o cinque elementi
   ii. il primo elemento sia un intero long
   iii. il terzo elemento sia un valore numerico reale formattato, per quanto attiene alla parte decimale, secondo le convenzioni italiane

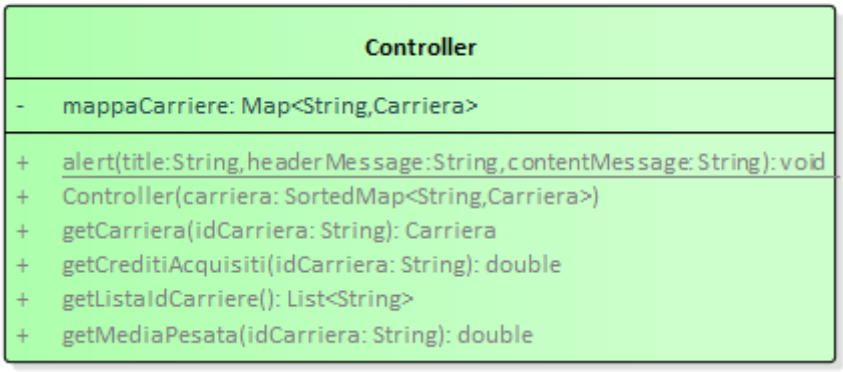
nonché, dove siano presenti anche gli altri due elementi:

   iv. la data sia correttente formattata secondo la struttura GG/MM/AAAA (vedi nota sopra)
   v. il voto sia un numero intero compreso fra 18 e 30, oppure una delle sigle sopra specificate

Parte 3 (punti: 7)

Package: mediaesami.controller *(punti 0)*

Il **Controller** (fornito) è organizzato secondo il diagramma UML nella figura seguente: esso mantiene internamente una mappa <stringa, carriera> che associa a ogni ***Carriera*** una opportuna stringa identificativa univoca.



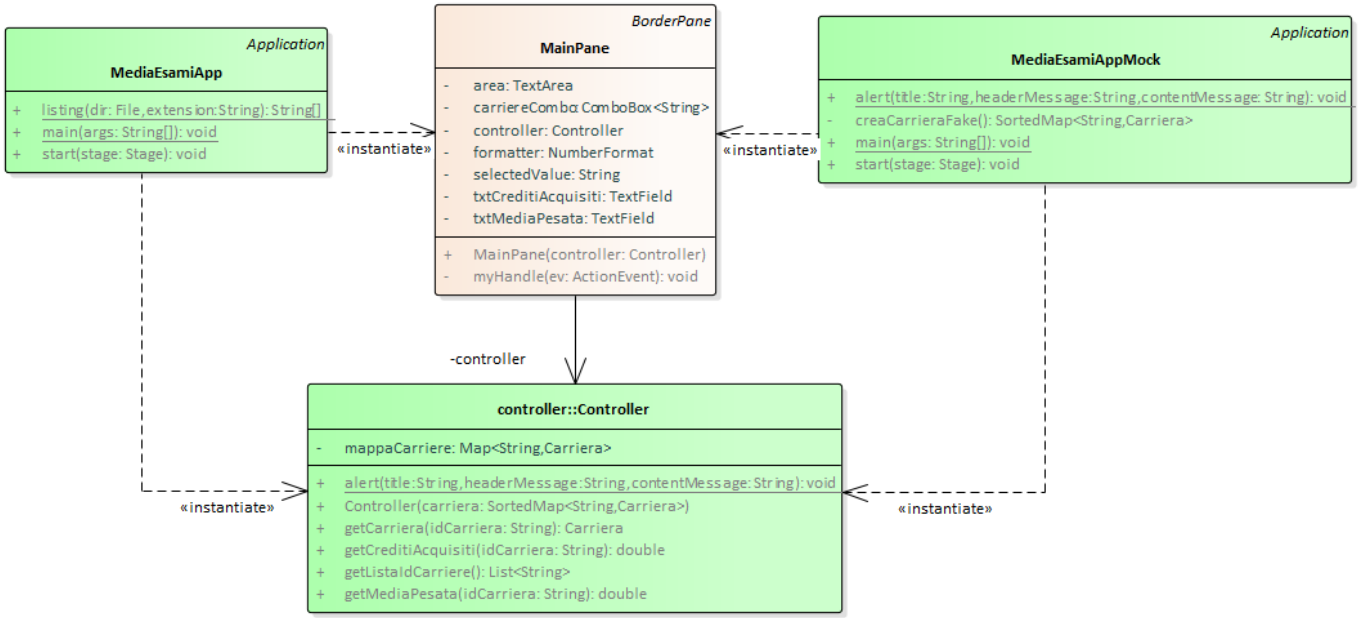
SEMANTICA:

- il costruttore riceve la mappa <stringa,carriera> su cui opererà
- il metodo *getCarriera* restituisce l'oggetto ***Carriera*** corrispondente alla chiave stringa ricevuta
- il metodo *getListaIdCarriere* restituisce una lista contenente le stringhe identificative delle varie carriere
- i due metodi *getMediaPesata* e *getCreditiAcquisiti* restituiscono rispettivamente la media pesata e i crediti acquisiti della carriera il cui identificativo (chiave stringa) è ricevuto come argomento
- la classe contiene anche il **metodo statico ausiliario *alert***, utile per mostrare avvisi all'utente.

Package: mediaesami.ui **[TEMPO STIMATO: 30-40 minuti]** *(punti 7)*

La classe ***MediaEsamiApp*** (fornita) costituisce l'applicazione JavaFX che si occupa di aprire i file, creare il controller e incorporare il ***MainPane***. Per consentire di collaudare la GUI anche in assenza / in caso di malfunzionamento della parte di persistenza, è possibile avviare l'applicazione mediante la classe ***MediaEsamiAppMock***.

L'interfaccia utente è illustrata nelle figure seguenti e segue il modello sotto illustrato*:*



L'interfaccia grafica si presenta come segue:

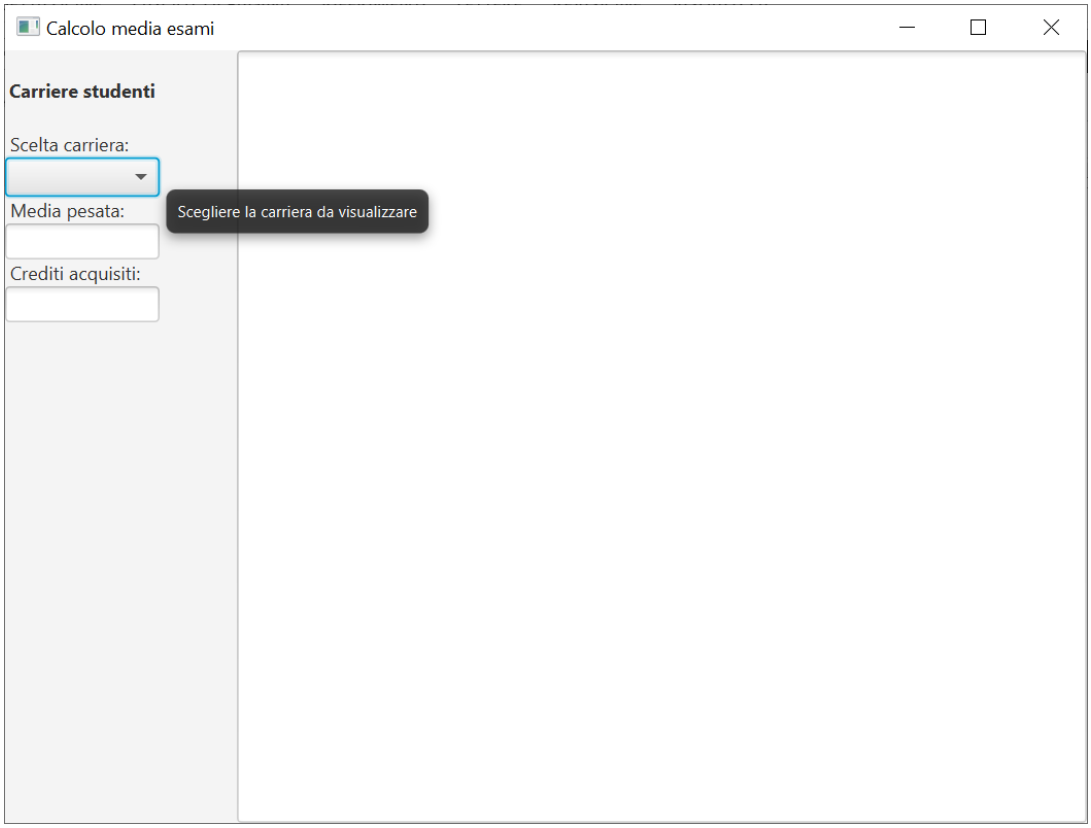
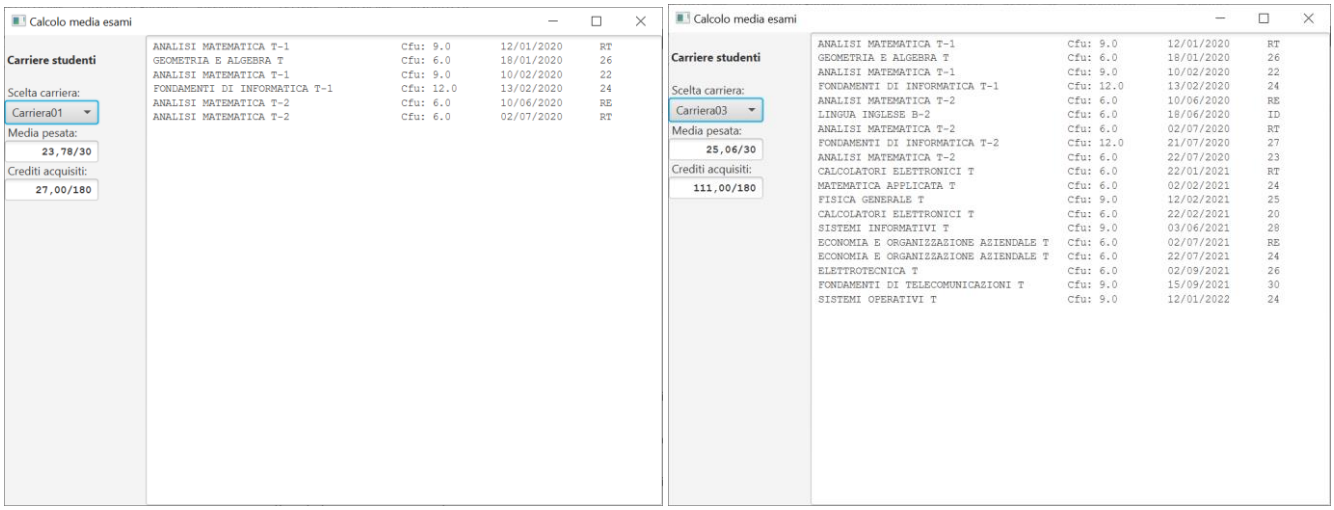


Fig. 1: la situazione iniziale della GUI, con combo vuota e nessuna selezione ancora effettuata

- a sinistra, una combo elenca le carriere disponibili; sotto, due campi di testo (inizialmente vuoti e non scrivibili dall'utente) mostrano rispettivamente le media pesata e i crediti acquisiti relativi alla carriera selezionata
- a destra, un'area di testo (inizialmente vuota e non scrivibile dall'utente) mostra i dettagli della carriera selezionata, ossia la lista degli esami in essa contenuti.

L'utente può interagire unicamente selezionando la carriera desiderata nella combo: immediatamente, i campi di testo sottostanti e l'area sulla destra vengono popolati con i dati corrispondenti



Figg. 2 / 3: la GUI dopo la selezione della prima carriera (a sinistra) e di una carriera successiva (a destra)

Il MainPane è fornito *parzialmente realizzato*: è presente buona parte dell'impostazione strutturale, mentre sono da completare la configurazione di alcuni componenti e la gestione degli eventi.

La classe ***MainPane*** (**da completare**) estende ***BorderPane*** e prevede:

1) a sinistra, una ***VBox*** con la combo e i due campi di testo, oltre alle opportune etichette
2) a destra, una ***VBox*** con la sola area di output.

La **parte da completare** riguarda:

1) la configurazione iniziale dei formattatori numerici
2) il popolamento della ***combo*** per la scelta della carriera
3) l'aggancio alla ***combo*** dell'opportuno listener incapsulato nel metodo ausiliario *myHandler*
4) la configurazione dei ***campi di testo*** non editabili che devono utilizzare il font Courier grassetto 11 pt
5) la configurazione dell'area di testo non editabile che deve utilizzare il font Courier (normale) 11 pt
6) la logica di gestione dell'evento, incapsulata nel metodo privato *myHandler*

In particolare, la gestione dell'evento deve:

- recuperare la carriera selezionata
- utilizzare tale dato per popolare i vari campi di uscita, tramite gli appositi metodi del controller

Cose da ricordare

- salva costantemente il tuo lavoro: l'informatica a volte può essere "subdolamente ostile"..
- in particolare: se ora compila e stai per fare modifiche, salva la versione attuale (non si sa mai)

Checklist di consegna

- Hai fatto un **JAR eseguibile**, che contenga cioè l'indicazione del main?
- Hai controllato che **si compili** e **ci sia tutto**? [NB: non includere il PDF del testo]
- Hai **rinominato** IL PROGETTO, lo ZIP e il JAR esattamente come richiesto?
- Hai **chiamato** la cartella del progetto esattamente come richiesto?
- Hai fatto un **unico file ZIP (NON .7z, rar o altri formati)** contenente l'intero progetto? In particolare, ti sei assicurato di aver incluso tutti i file .java (e non solo i .class)?
- **Hai consegnato DUE file distinti, ossia lo ZIP col progetto e il JAR eseguibile?**
- Su EOL, hai **premuto** il tasto "CONFERMA" per inviare il tuo elaborato?